Prediction Assignment

Lynna

Friday, November 21, 2014

Background

The goal of this assignment is to use the data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants, to create a model that can predict the manner in which the exercised were done. More information about the data can be found on this website: http://groupware.les.inf.puc-rio.br/har.

Data loading and cleansing

The data includes 160 variables, one of which is the dependent variable, "classe". The manner of exercise was recorded under this variable. It's a factor variable with 5 levels, which are labeled with letters A to E.

The data contains many missing values, which are labeled with "NA". Any columns with too many NA's are useless, so the for-loop below is used to find the amount of NA's in each column. There are 60 columns which do not have NA's, so these are kept and the rest are dropped. The first 7 columns are also dropped since they are just identifier varibles such as time and username.

```
setwd("C:/Users/lynnaj/Desktop/Desktop/coursera/machine learning/assignment")
dataTraining <- read.csv("pml-training.csv", na.strings= c("", " ", "NA"))

NAdf <- NULL
for(i in 1:ncol(dataTraining)){
  NAdf[i] <- sum(is.na(dataTraining[,i]))
}

dataTraining <- dataTraining[,which(NAdf == 0)]  #keep only the variables which has no NA's

dataTraining <- dataTraining[-c(1:7)] #delete first 7 columns as they're not applicable predictors.
```

After the data cleansing is finished, there are 53 variables left.

Model

Cross validation can be performed now with the clean training data, which gets split into 70/30. The 70% of the clean training data is used to build the model. Then the other 30% is the cross validataion test set.

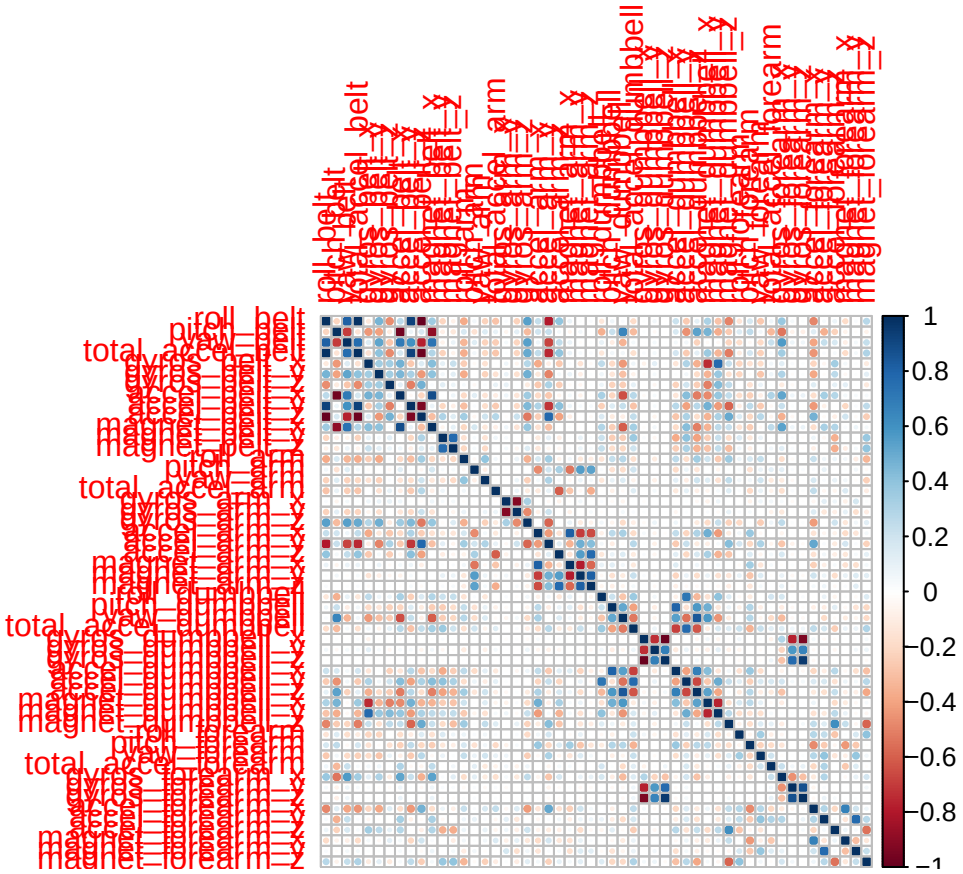
Random Forests method is chosen for modeling building, because it can handle thousands of input variables without variable deletion. Plus, it is suitable for such a classification problem.

```
library(randomForest)
library(corrplot)
library(caret)
library(corrplot)
set.seed(123)
```

```
#cross validation
inTrain <- createDataPartition(y = dataTraining$classe, p = 0.7, list = FALSE)
trainset <- dataTraining[inTrain, ]
crossset <- dataTraining[-inTrain, ]
```

In order to reduce the forest error rate, the correlation between the independent variables are explored. By observing the correlation plot graph, the variables which have high correlation with others are picked out. Three variables are picked to be deleted. They are "accel_belt_x", "roll_belt", and "total_accel_belt".

```
correlations <- cor(trainset[, -c(53)])
corrplot(correlations)
```




```
trainset <- trainset[-c(1, 4, 10)]  #delete the 3 variables

modelFit <- randomForest(classe ~ ., data = trainset)
modelFit
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = trainset)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
```

```
##         OOB estimate of  error rate: 0.54%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3903    1    0    0    2    0.000768
## B   11 2641    6    0    0    0.006396
## C    0   17 2376    3    0    0.008347
## D    0    0   25 2226    1    0.011545
## E    0    0    3    5 2517    0.003168
```

The results of the model report that the out-of-bag (oob) error is 0.54%. And when the model is used to predict the cross validation test set, the results report the accuracy is 99.4%.

```
# crossvalidate the model using the remaining 30% of data
predictCross <- predict(modelFit, crossset)
confusionMatrix(crossset$classe, predictCross)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    1    0    0    0
##          B    6 1132    1    0    0
##          C    0   13 1013    0    0
##          D    0    0   13  950    1
##          E    0    0    0    0 1082
##
## Overall Statistics
##
##                Accuracy : 0.994
##                  95% CI : (0.992, 0.996)
##     No Information Rate : 0.285
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.992
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.996    0.988    0.986    1.000    0.999
## Specificity             1.000    0.999    0.997    0.997    1.000
## Pos Pred Value          0.999    0.994    0.987    0.985    1.000
## Neg Pred Value          0.999    0.997    0.997    1.000    1.000
## Prevalence              0.285    0.195    0.175    0.161    0.184
## Detection Rate          0.284    0.192    0.172    0.161    0.184
## Detection Prevalence    0.284    0.194    0.174    0.164    0.184
## Balanced Accuracy       0.998    0.993    0.992    0.999    1.000
```

The model performs well under cross validation, so it is used on the test set.

Prediction

By using the model, we can predict the observations included in the test set. Here are our predictions:

```r
# load the test data with 20 observations
dataTest <- read.csv("pml-testing.csv", na.strings= c("", " ", "NA"))

# predict the classes of the test set using the model
predictTest <- predict(modelFit, dataTest)
predictTest
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  B  A  A  E  D  B  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```